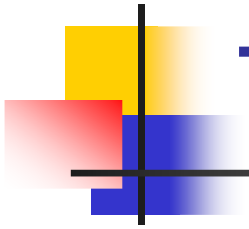
FOBS: Fast Object Based Data Transfer System

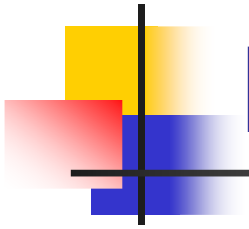
A scalable, portable, lightweight protocol for data transfer on the Grid

Vinod Kannan
http://kannvin.freeshell.org
saranga2000:@yahoo.com



The Problem:

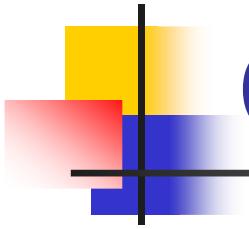
- Computational Grids
- Upto 40 gbps Bandwidth (TeraGrid)
- Applications are unable to utilize the installed bandwidth effectively
- Reason: TCP



Problems with TCP

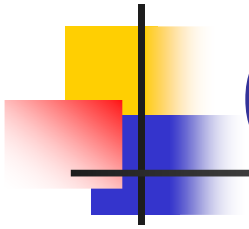
- Window-based congestion control mechanism and the size of the window.
- Aggressive (AIMD) congestion control mechanism, which is designed for the Internet1.
- Slow start mechanism
- Acknowledgment scheme in basic TCP

Window Based Congestion Control



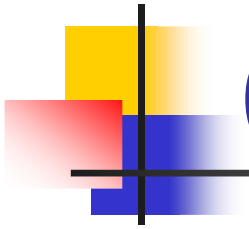
- Flow and congestion control windows
- To ensure sender doesn’t overwhelm receiver buffer or use bandwidth unfairly
- Flow-control window (fwnd) and a congestion control window (cwnd).
- Effective window (ewnd) = min (fwnd, cwnd)

Window based mechanism (contd)

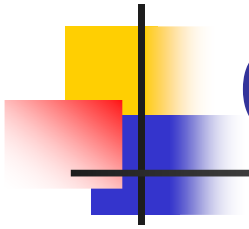


- Cwnd varies dynamically as the network state changes, while fwnd is always static.
- Ideally, fwnd should vary with the bandwidth delay product (BDP), a measure of the amount of data in transit
- On most operating systems, fwnd is set to 64 KB without scaling
- BDP can vary from a few bytes (56 Kbps * 5 ms = 36B) to a few megabytes (1000 Mbps * 100ms = 7.8 MB)

Window based mechanism (contd)

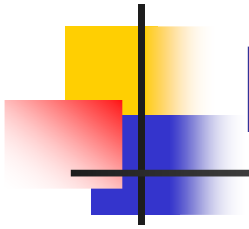


- Thus, in the first case, over 99% of the memory allocated is wasted, while in the latter, 99% of the network bandwidth is wasted
- Some solutions are increasing the window size. Need root permission.



Other problems exist too

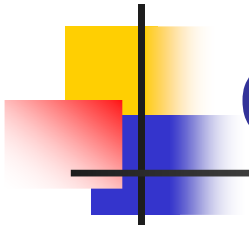
- AIMD Congestion control.
- Self similar (chaotic) nature of TCP traffic.
- Acknowledgment for each and every segment transmitted.



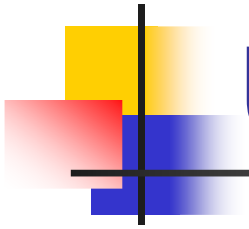
Proposed Solutions

- Two approaches
  - Overcoming inherent drawbacks of TCP
    - Eg. TCP Vegas, SACK, High-performance TCP implementations
  - Application level mechanisms
    - Circumvent TCP drawbacks by data “striping”
    - UDP based approaches

Overcoming Inherent drawbacks in TCP



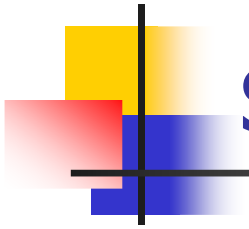
- Increasing fwnd and cwnd: Needs root access
- Selective acknowledgment
- “Fast Transmit” and “Fast Recovery” mechanisms
- Alternate Congestion Control mechanisms
- Need root access, and dynamic tuning for different links
- http://www.psc.edu/networking/perf_tune.html



User-Level Approaches

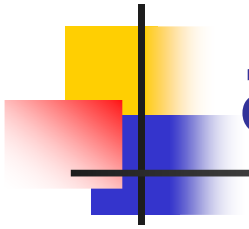
- Two approaches
- Research aimed at circumventing the drawbacks of TCP.
- Uses multiple TCP streams to “stripe” data across many streams.
- Eg. PSOCKETS, GridFTP, BBCP, BBFTP
- Improved performance as TCP limitations are on a per-stream basis.

Problems with Multiple streams



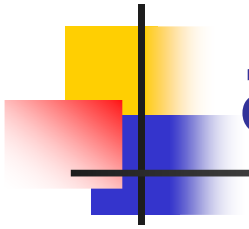
- Aggregate TCP buffer size close to the BDP.
- Circumvent Congestion control mechanisms because always some stream is ready to "fire"
- Kernel costs associated with maintainng multiple streams
- Problems with finding the optimal number of streams

UDP based user level approaches



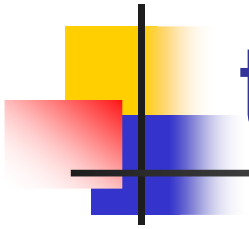
- Eg. RUDP, Tsunami, Sabul.
- Reliable Blast UDP: (EVL UIC)
- How it works.
- Problems
  - No Congestion control
  - Designed for networks with very low probability of packet loss.

UDP based user level approaches (contd)

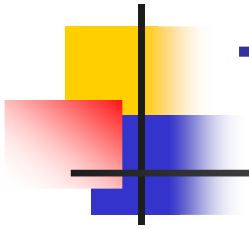


- Tsunami (ANML Indiana University)
  - Experimental (Repeatedly Seg faulted in user trials)
  - Very aggressive in utilizing bandwidth
- SABUL (LAC- UIC)
  - Highly system and architecture specific
  - Congestion control closely tied to that of TCP

FOBS: Fast object based data transfer system

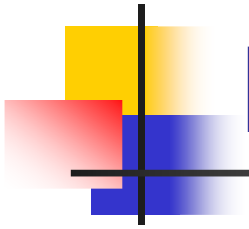


- Application level data transfer mechanism
- Uses UDP for data transfer
- Performance implications of using UDP (Overhead in TCP segments), decrease from 20 bytes to 12 bytes
- Unreliable, so application must provide the acknowledgment and retransmission mechanism



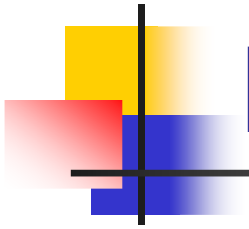
Terms associated with FOBS

- CHUNK
- Segment
- Data packet
- Feedback or Acknowledgment packet



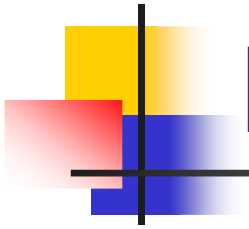
Mechanism of FOBS

- Data over UDP
- Feedback over UDP/TCP
- Acknowledgment and retransmission mechanism
- Rate Control
- Congestion Control



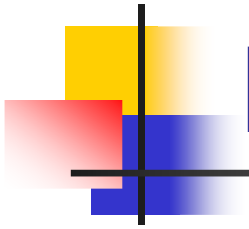
FOBS: Initial Design

- Initial Design: Mechanism of FOBS
- Byte-Acknowledgments over UDP
- Byte-Acknowlegdment- one byte per packet
- Acknowledgment was on per-chunk basis, and overlapping.
- Too big (eg. 40 MB chunk = 27 Kbyte byte ack) to be fit into a MTU
- What is MTU?
- Not scalable



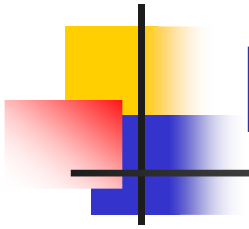
FOBS: Initial Design (contd)

- Bit-Acknowledgment
- Still too big (100 MB chunk = 9 Kb bit ack)
- Translation costs
- Overlapping nature
- Loss of acknowledgments sent over UDP
- Lack of Rate Control: Problem while sending from “Fat” to “Thin” Pipes
- Lack of Congestion Control



Re-designed FOBS

- Segmented Approach
- Segmented Bit-Acknowledgment
- Acknowledgment size de-coupled from Chunk size
- Segment size chosen optimally.
- Ex. 10,000 packet segment = 1264 byte acknowledgment
- Can be sent over TCP

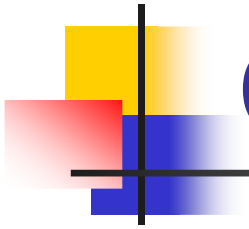


Drawbacks with Segmented Bit-Acknowledgments

- Costs in translation and transmitting over TCP
- Given the nature of the networks, little or no congestion.
- Need for suite of acknowledgment schemes

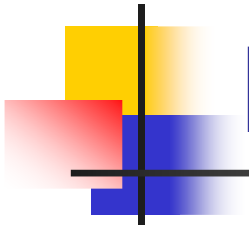
Sample Acknowledgment Sizes

Ack Type for 100 MB Chunk	Byte	Bit	Segmented Bit Feedback (seg Size 10,000) packets	Segmented Ack	NACK
Size in bytes	69 Kb	8.5 Kb	7segments * 1264 bytes	7 *12 bytes	<1500 bytes



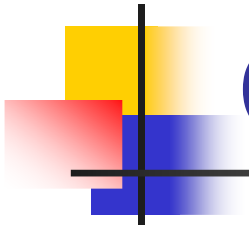
Composite Acknowledgments

- Segmented Composite Acks
- ACK – Selective Positive Acknowledgment for a segment (12 bytes)
- NACK- Negative Acknowledgment (< MSS) for whole chunk
- Significant improvement but cutting down processing costs



Rate Control in FOBS

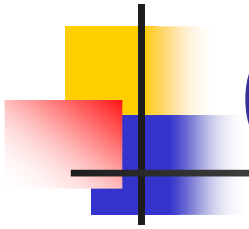
- Rate Control: Need for rate control
- “Fat” to “Thin” pipe transfer
- Ideal Rate control implementation
- Overheads involved, system clock granularity
- Rate control for a batch of packets



Congestion Control in FOBS

- Vital as the Grid is shared
- State based congestion control.
- Changes in the behavior of the system are based on the current state of the network gained through the feedback information
- Implementation Explained (next slide)

Congestion Control in FOBS (contd)



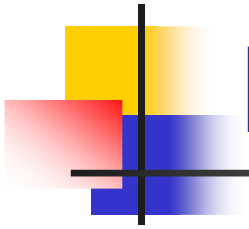
- 3 States: GREEN, YELLOW and RED denoting clear, moderately congested and heavily congested network states
- Each state has maximum and minimum sending rates
- Distinct set of conditions for state change
- Calculation of available bandwidth from the loss percentage.
- Moving from one state to another.

Congestion Control in FOBS (contd)

State	Lower Boundary (Mbps)	Mid Boundary (Mbps)	Upper Boundary (Mbps)	Increase Steps (Mbps)	Decrease Steps (Mbps)
GREEN	85	92	100	3	3
YELLOW	20	52	84	5	8
RED	1	10	19	1	3

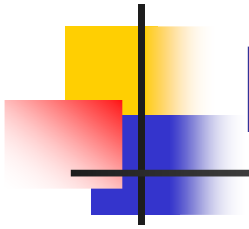
Congestion Control in FOBS (contd)

Initial State	New State	New Sending rate for the state change (Mbps)
RED	YELLOW	20
RED	GREEN	92
YELLOW	GREEN	100
GREEN	YELLOW	84
GREEN	RED	1
YELLOW	RED	10



Portability Issues

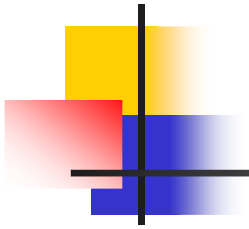
- Using POSIX standard functions
- How Other protocols (SABUL) are able to obtain marginally better performance
  - User buffer Registration to avoid memcopy
  - High Granularity architecture dependent clocks



Experiments

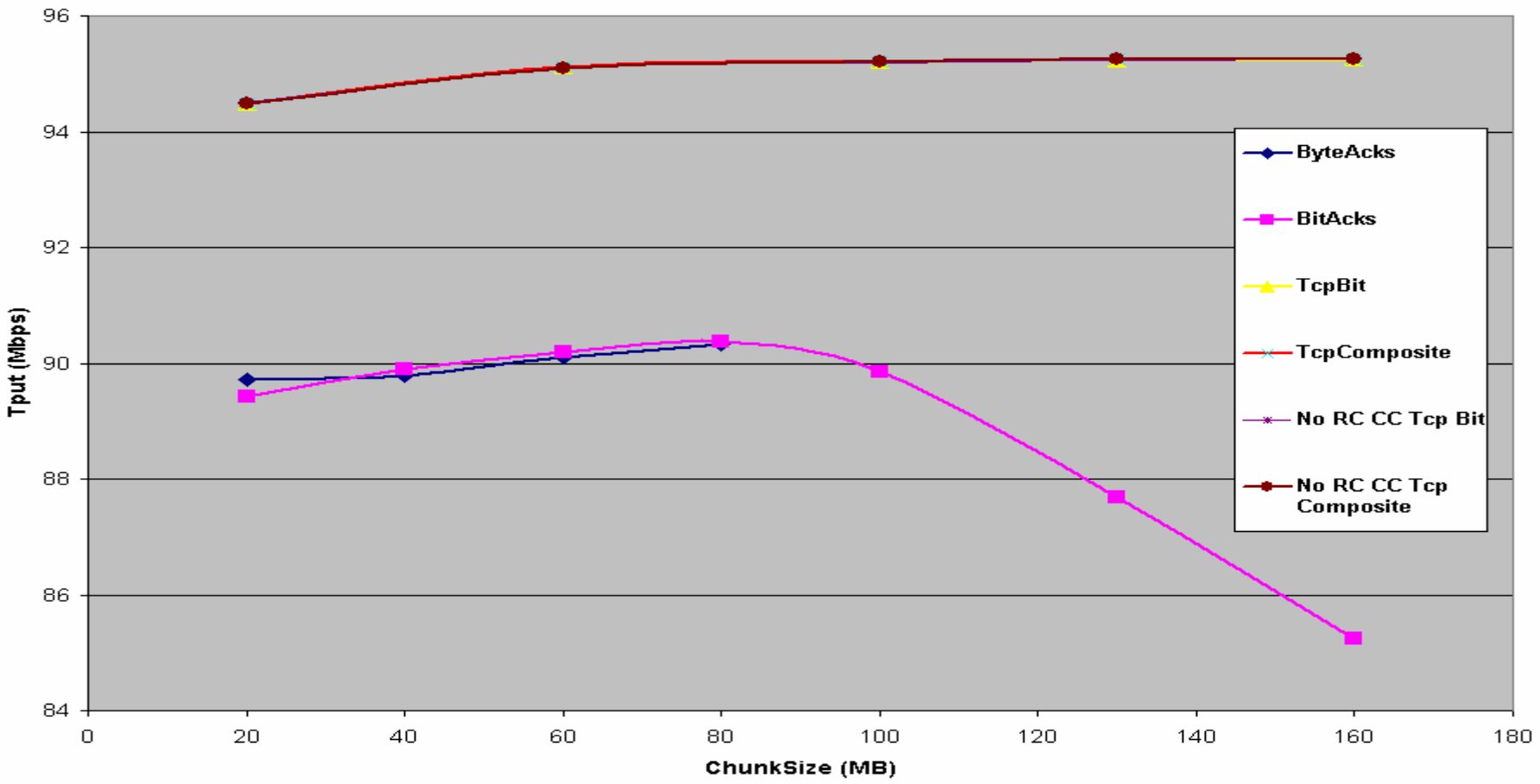
- Experimental Setup: Host Configurations

Host	Host Type	CPU's (used only one CPU)	Network Connection	Operating System
ANL	Linux box	2 * 500 MHz Pentium III	100 Mbps	RedHat Linux 6.1
NCSA	SGI Origin 2000	64 * 195 MHz R10000	100 Mbps	IRIX 6.5
CACR	HP V2500	64 * 400 MHz PA-8500	660 Mbps	HP-UX 11.10

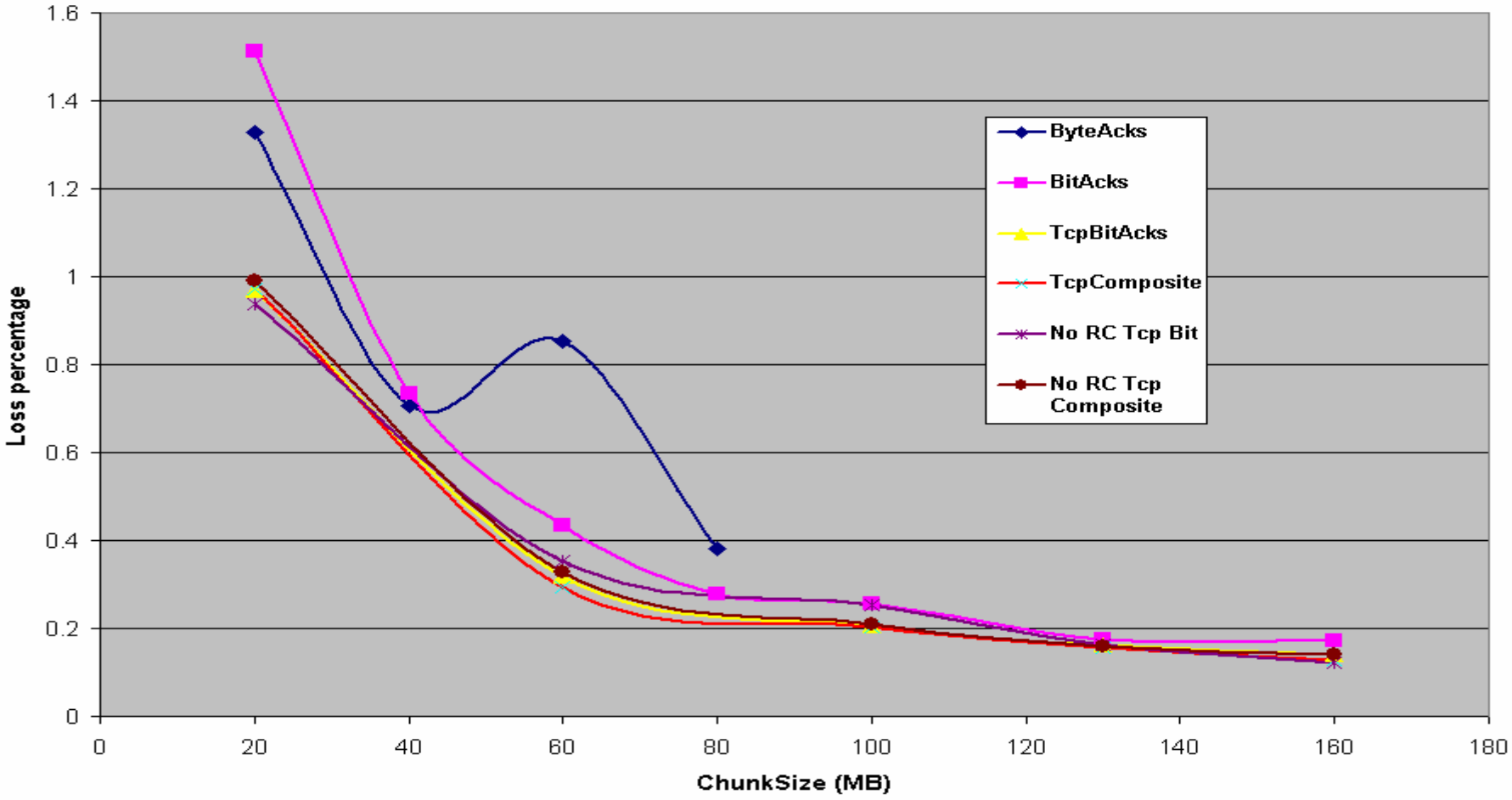


- Short Haul
  - ANL to NCSA
  - NCSA to ANL
- Long Haul
  - NCSA to CACR
  - CACR to NCSA

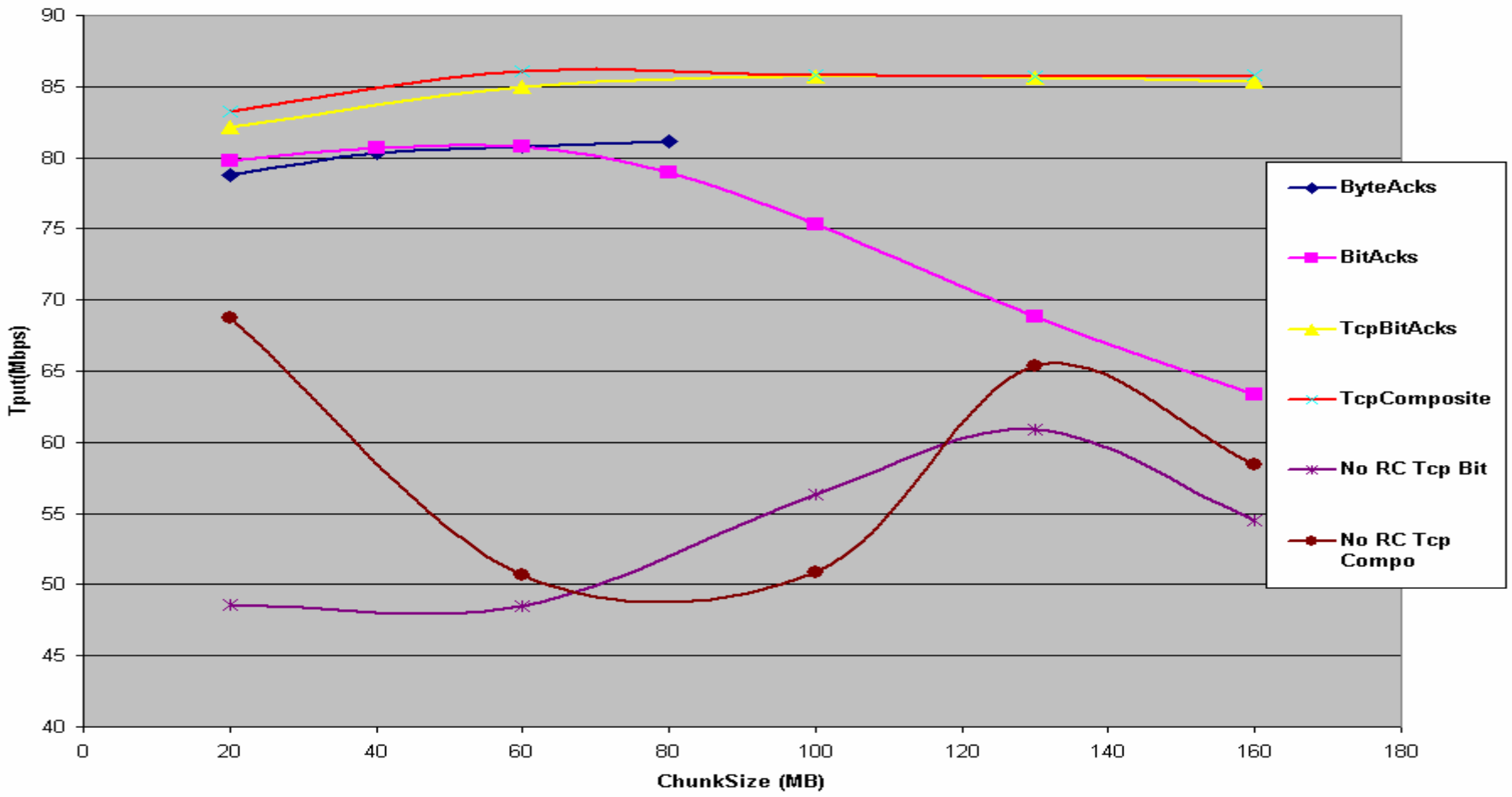
ANL to NCSA: Throughput



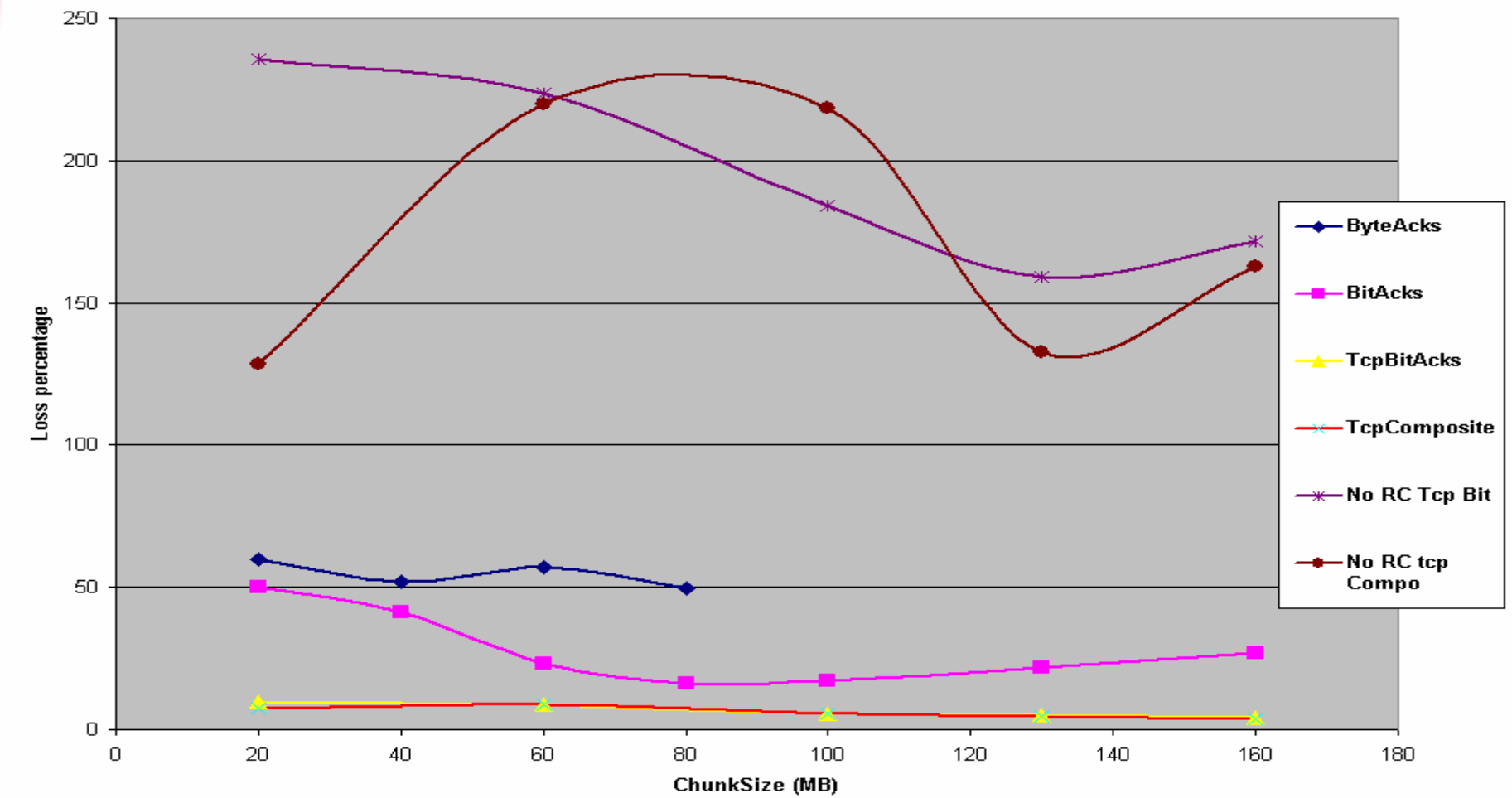
ANL to NCSA: Loss percentage



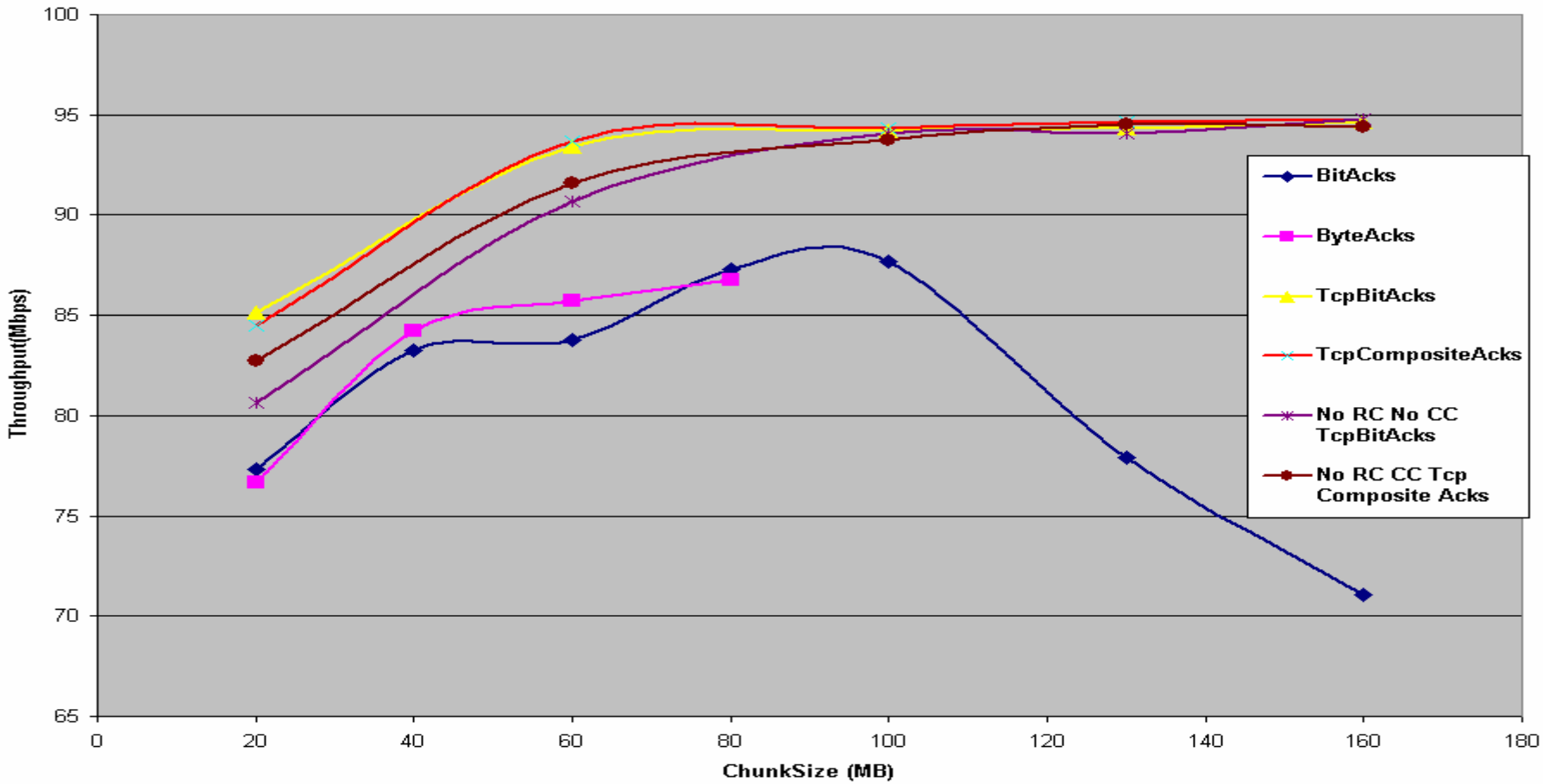
NCSA to ANL: Throughput



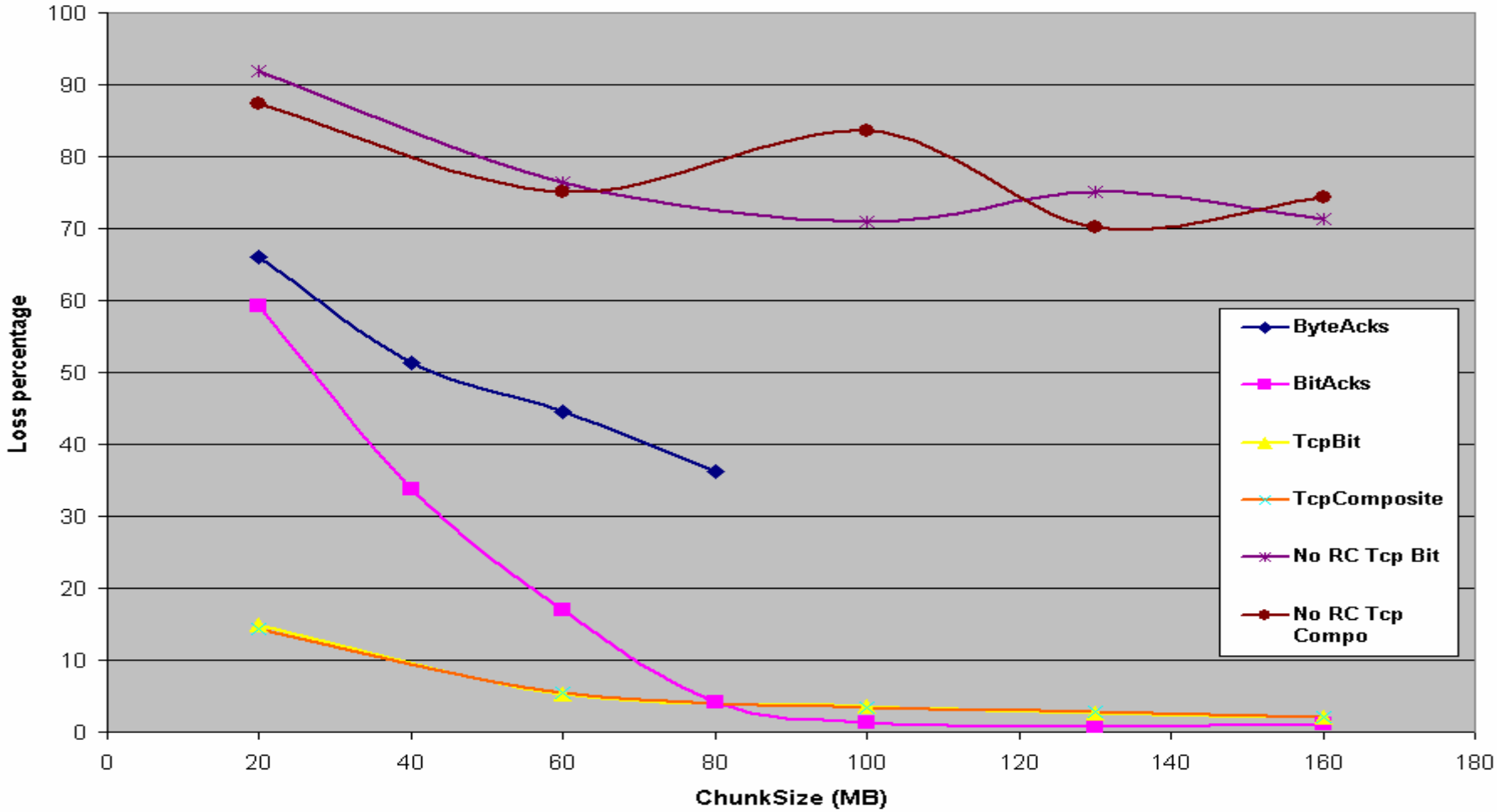
NCSA to ANL: Loss Percentage



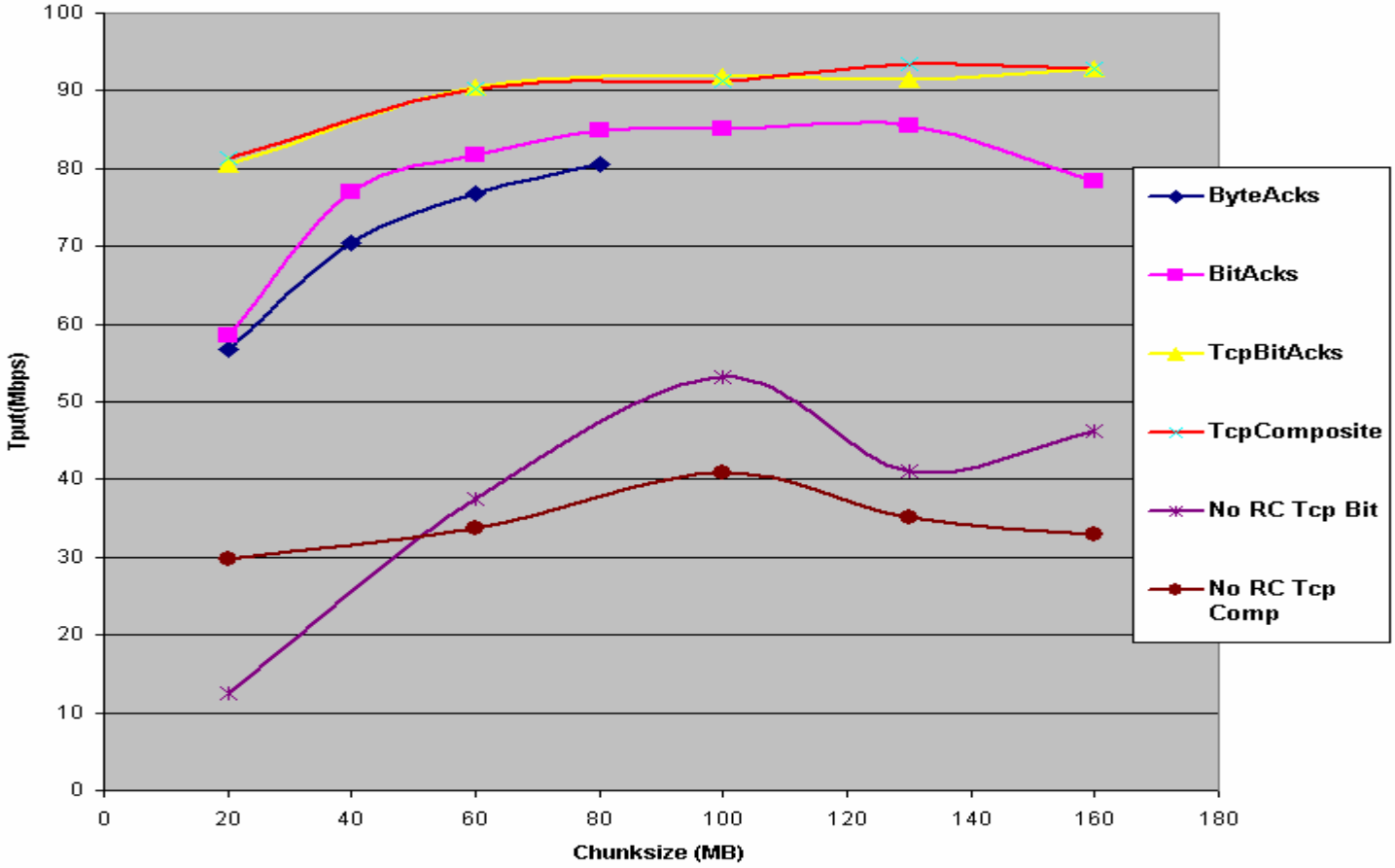
NCSA to CACR: Throughput



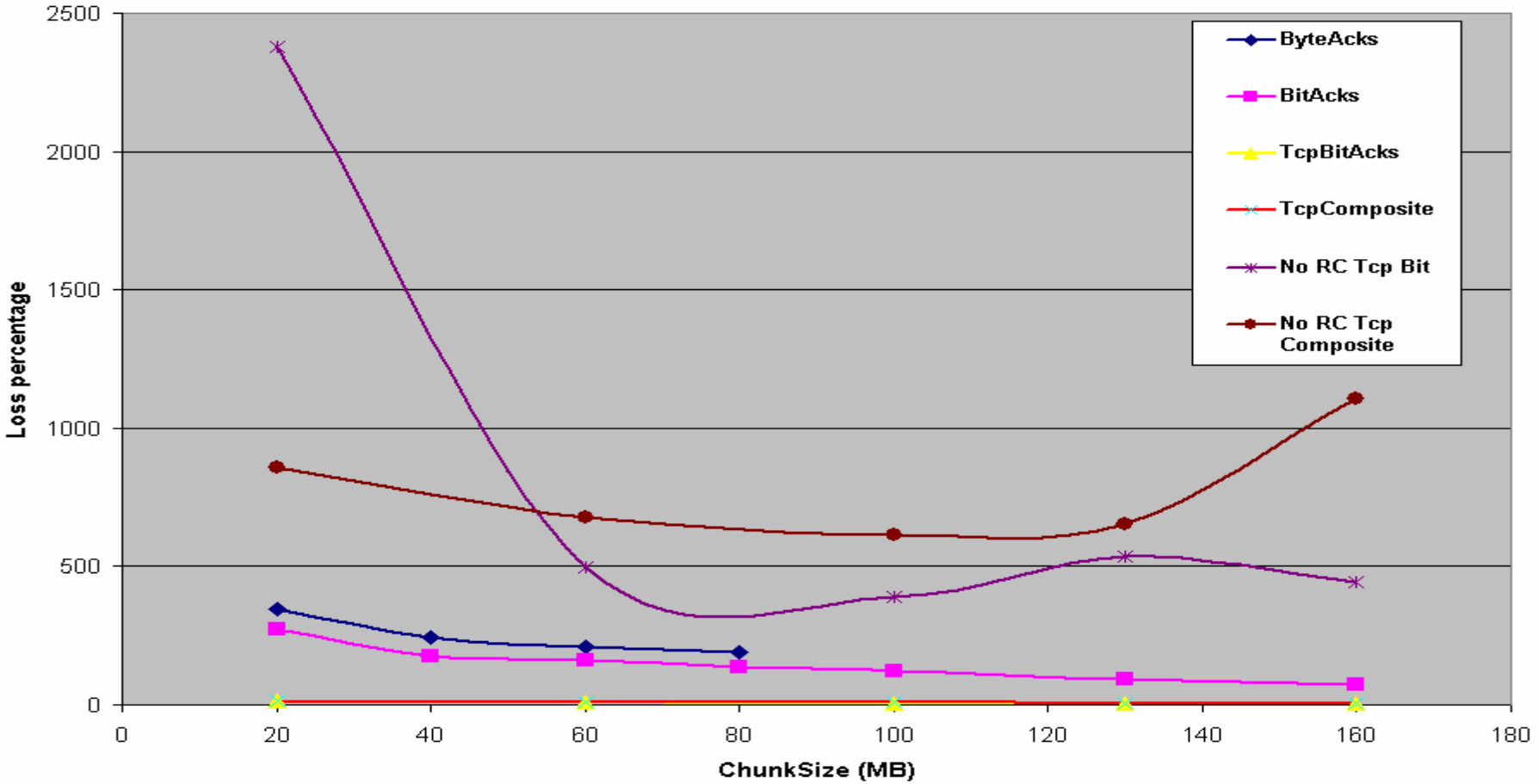
NCSA to CACR: Loss Percentage

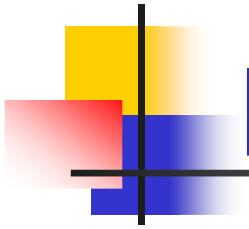


CACR to NCSA: Throughput

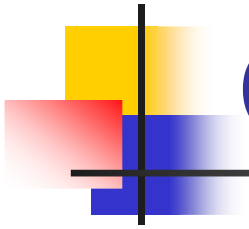


CACR to NCSA: Loss %age



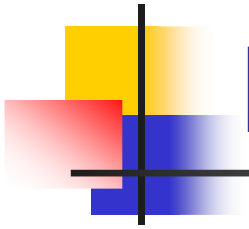


Comparision with other protocols



Conclusions

- Achieves up to 95% of bandwidth (95 Mbps on Fast Ethernet) and 580 Mbps on OC12
- Loss percentage kept within acceptable, user specified boundaries.
- Portable and tested across all major platforms
- Easy User Interface (scp like command line and Java GUI both using a fobs daemon)



Future Work

- Alternate Congestion control models
- Fine tuning FOBS for gigabit links